July, 1976

Report ESL-IR-677

Contract ONR/N00014-75-C-1183

12

DATA NETWORK RELIABILITY - FIRST ANNUAL REPORT
1 JULY 1975 - 30 JUNE 1976

John M. Wozencraft
Principal Investigator

Electronic Systems Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY, CAMBRIDGE, MASSACHUSETTS 02139

Department of Electrical Engineering and Computer Science

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) UNCLASSIFIED

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) DATA NETWORK RELIABILITY. (FIRST ANNUAL REPORT)		5. TYPE OF REPORT & PERIOD COVERED Annual, Jul 75-Jun 76
		6. PERFORMING ORG. REPORT NUMBER (14) ESL-IR-677
7. AUTHOR(s) (10) R. G. Gallager, A. Segall, John M. Wozencraft		8. CONTRACT OR GRANT NUMBER(s) ARPA Order No. 3045/5-7-75 ONR/N00014-75-C-1183
9. PERFORMING ORGANIZATION NAME AND ADDRESS Massachusetts Institute of Technology Electronic Systems Laboratory Cambridge, Massachusetts 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Code No. 5T10 ONR Identifying No. 049-383
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209		12. REPORT DATE (11) July 1976
		13. NUMBER OF PAGES 61
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Office of Naval Research Information Systems Program Code 437 Arlington, Virginia 22217 (12) 61p.		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

(9) Annual rept. no. 1, 1 Jul 75 - 30 Jun 76

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

(15) N00014-75-C-1183, W ARPA Order-3045

18. SUPPLEMENTARY NOTES

(16) NR-049-383

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Data	Routing
Communication	Adaptability
Network	Protocol Information
	Resource Sharing

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The objective of this research is how to achieve in engineering practice the potential advantages of reliability that a large scale data communication network should make possible. From a technical viewpoint, the principal areas of investigation reported here are computer resource sharing, adaptive network routing, and the efficient representation of network protocol information. The approach has been theoretical, with an attempt made to take a fundamental new look (independent of current engineering practice) at areas where theoretical over

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

127 200

LB

20. Continued

understanding is limited, but important and possible to attain. The primary analytical tools have been information, control, and queueing theory.

ESL-IR-677

JULY 1976

FIRST ANNUAL REPORT

(1 July 1975 - 30 June 1976)

For the Project

DATA NETWORK RELIABILITY

Research Supported by

INFORMATION PROCESSING TECHNOLOGY OFFICE

DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

DEPARTMENT OF DEFENSE

Principal Investigator:
John M. Wozencraft
(617) 253-7269

ARPA Order No. 3045/5-7-75
Program Code No. 5T10

ONR/N00014-75-C-1183
ONR Contract No. 049-383

Electronic Systems Laboratory
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Table of Contents

DATA NETWORK RELIABILITY

1. Introduction

1.1 Research Objectives

The root question to which our research is addressed is how to achieve in engineering practice the potential advantages of reliability that a large scale data communication network should make possible. From a military point of view, three of these potential advantages seem most critical:

- An increased capability to disperse user facilities geographically; to reduce the fraction of individual user resources concentrated in one area; and to replicate these resources efficiently in different areas.
- Reduced dependence on the proper functioning of individual network switching installations and communication links, through the availability of many alternate routes.
- Increased surge capacity for high priority traffic under emergency conditions, combined with efficient utilization of network resources for standard traffic mixes under normal circumstances.

Stated briefly, the corresponding technical objectives are resource sharing, adaptability, and efficiency.

1.2 Research Approach and Phasing

Data communication networks are already an engineering reality. One approach to enhancing their reliability would therefore be to search out ways to improve existing practice. But it often happens that engineering practice leaps ahead of a basic understanding of underlying issues. To a large extent this seems true of the current state of data networks. Our initial approach has therefore been to focus on fundamental areas where theoretical understanding seems limited, but important and possible to attain. Specific areas which have been addressed during the past year include message protocol information, adaptive routing, and dynamic file allocation. Significant results have been obtained in several of these studies and are considered at some length in the body of this report.

It should be emphasized, however, that our main goal during this first year's work has been to develop new analytical tools and new conceptual insights adequate for dealing with the problems of network reliability. Most of our work (particularly where doctoral thesis research is concerned) is not expected to culminate for another year, and even those results we have already obtained often raise as many questions as they answer. Accordingly, in the body of the report we also discuss the prospective work to which last year's efforts have led us.

Although the conceptual foundation on which our research approach is based must still be regarded as unproven, a brief overview of our current philosophy regarding network reliability should help keep the more detailed discussions that follow in perspective. We believe that there are two main cornerstones:

(1) In order for networks to be reliable, their control must be adaptive, distributed, stable, and deadlock-free.

(2) In order for networks to be economically affordable, they must be compatible and efficient.

Clearly, the engineering difficulty lies in achieving all attributes simultaneously, and the antecedant intellectual difficulty lies in quantifying the trade-offs among them.

1.3 Organization of Report

The tightly intertwined relationship among various apparently disparate aspects of data network reliability has led us to adopt a somewhat unusual organization for the report. In Section 2 we introduce a network taxonomy which provides a skeleton around which our work to date is summarized. The treatment in that section is discursive in nature, and longer than summarization alone would have required. The additional material expands on what the key issues of network reliability (as we see them) are, and serves to explain the rationale behind the specific topics that have been studied.

Section 3 treats individual research topics. The order of presentation is the same as in Section 2, but the level of detail is greater and the focus is more on the specific work than (as in Section 2) on the context into which it fits. Prospective future work is outlined at the end of each subsection.

To the maximum extent possible, we have tried to make this report of our first year's efforts self-contained and readable. Emphasis has been placed on the nature and anticipated significance of the results, rather than on their derivations, which can be found in the references.

2. Summary of Work Accomplished

The overall subject of data network reliability can be looked at from so many different point of view that it is difficult to discuss (or even to study) it without imposing some structure in terms of which specific sub-problems can be related. The taxonomy we have adopted decomposes an overall computer/communication network into four layers, namely the

computer subnet, the
computer/communication interface, the
communication subnet, and the
communication links,

each of which can be addressed with some degree of independence. It should be emphasized from the outset, however, that major interdependencies, particularly insofar as reliability is concerned, are bound to remain regardless of the taxonomy adopted.

2.1 Computer Subnet

A principal function of a data communication network is to provide facilities for the sharing of resources in such a way that time-varying demands for computer service may be accommodated reliably and efficiently. Many of the problems involved here relate to computer system architecture, and exist (especially in the case of time-sharing and multiprocessor systems) even when the communication delays between system modules are negligible relative to the data processing delays. The imposition of significant communication delays, however, introduces additional problems, and it is on these that our work has focused.

2.1.1 File Allocation

One way in which a geographically dispersed computer network can impact on reliability involves the common use of data bases and information files by all computers in the system. When a file is used by several computers in the network, it can be stored in the memories of one or more of them, and accessed by others via the communication channels. In an environment where computers communication links can fail because of technical reasons or military action, the topology of the network is changing dynamically, and dynamic allocation of the files is therefore necessary. A specific question we have addressed is how the number and location of copies of files in the network should be varied as a function of the required reliability, as well as of such parameters as failure and recovery probabilities and the costs of file storage and transmission.

2.1.2 Load Sharing

Another network impact on reliability involves load sharing: if the performance of a subset of the computers in a system degrades (catastrophically or in part), the overload can be redistributed throughout the system. We have investigated how to effect this redistribution in such a way that the average total job processing-plus-communication time is minimized, as a function of the topological distribution of job arrivals and of the available computational and communication capabilities.

2.2 Computer/Communications Interface

We have not yet begun substantive work on problems related to the computer/communications interface, but expect to do so during the coming year. The discussion that follows is intended to establish how interface (specifically, flow control) problems relate to other reliability issues in general, and to our other work in particular.

The place where unreliability in a network manifests itself to users is at the computer/communications subnet interface. Given that messages must not be lost, the two most evident types of failures are long delays caused by congestion, and network refusal to accept new messages. It is a tautology that there is a tradeoff between the two, since congestion could never occur in a network that refused all messages.

The set of strategies employed in a communication network to prevent new traffic from entering the network when there is danger of congestion is called flow control. Congestion, in turn, refers to situations in which packets or messages are rejected or discarded at intermediate nodes because of lack of buffer space. Taken together, congestion, flow control, and the recovery procedures used when congestion occurs pose two somewhat different kinds of threats to the reliability of a network, namely deadlocks and instability.

Deadlocks are situations in which a set of events must occur before the network can proceed with its business, but in which each event must await the occurence of another event in the set; since no event can occur first, the functioning of the network (or some part of it) is brought to a halt. The conventional approach to deadlocks is to recognize their existence, usually by a timeout (for example, a maximum specified interval between message acceptance and acknowledgement) and then to institute a recovery procedure. Typically the timeout and recovery procedure, coupled with the other asynchronously occuring events in the network, will generate new more subtle (but hopefully less likely) deadlocks. As far as we know, there are no networks in existence which one could assert to be deadlock free with much feeling of confidence.

Instability is the phenomenon by which message delay increases with throughput up to a certain point and then continues to increase while throughput decreases. When congestion occurs, the increased communication and processing loads, due to retransmitting rejected messages, can keep the network in a congested state even if the inputs to the network have dropped off to a point where congestion would not normally occur.

Both routing and flow control play a role in preventing congestion. We regard flow control as being in a different layer from routing (which we relegate to the communication subnetwork) partly because flow control is exercised at the input to the communication network, and partly because current flow control techniques are end-to-end procedures to which the routing is more or less transparent. During the past year we have emphasized work on routing, rather than on flow control, in the belief that traffic should be refused only when less drastic recourses become ineffectual.

The issue of flow control is particularly critical in internetworking, since when the flow control of a recipient subnetwork prevents messages from entering it, the danger of congestion in the donor subnetwork increases.

This raises the question of the extent to which flow control should be regarded as a global issue in a network of networks and the extent to which the subnetworks can have their own flow control.

2.3 Communication Subnet

The main responsibilities of the communication subnet are twofold:† first, to accept messages from their source and deliver them expeditiously to their destination; and if unable to do so, then second to institute recovery procedures which maintain message integrity (i.e., no loss of messages) at a minimum cost in lost time.

2.3.1 Recovery

Like flow control, recovery procedures are tightly dependent on message routing and have not yet been explicitly studied by us. Our prospective work in this area is discussed in Section 3.3.1.

2.3.2 Routing

The reliability of the communication subnet depends critically both on network topology and the algorithms used to determine the routes which messages follow from their source to destination. Routing algorithms can be classified roughly as static, quasistatic, and dynamic. Static algorithms are usually designed to determine fixed routing tables or network flows to minimize some statistic (usually the average) of network delays, assuming stationary input requirements. Such algorithms are used both in network design and in setting up fixed routing tables. Quasistatic algorithms are similar, except that they are specifically structured to adapt to slow variations in the input requirements and in the network topology. Finally, dynamic routing algorithms choose routes for individual messages or packets based on available information about queues in the network and other status variables.

In a military environment, communication link capacities can fluctuate widely as a result of jamming or physical destruction, and a processor

---

†In our taxonomy, responsibility for maintaining the accuracy of each message falls upon the communication links.

responsible for calculating routing tables may be destroyed. Thus it is crucial that routing algorithms should be at least quasi-static, and highly preferable that they be distributed (i.e., not dependent upon any particular node or subset of nodes for proper functioning). The ARPANET, for example, has both these attributes.

Adaptive routing policies are important also when link capacities and network topology are unchanging. This is because the network input traffic will still vary randomly, and congestion can occur as readily from a statistical doubling of traffic requirements as from a halving of network resources. From the user's standpoint, a network that refuses to accept his messages a significant fraction of the time is unreliable. From an engineering standpoint, what is needed to avoid this kind of unreliability on the one hand, or uneconomic overdesign on the other, is a basic understanding and quantification of "best possible" routing performance to serve as a standard against which network congestion/thruput tradeoffs attainable with feasible procedures can be measured. The problem is independent of whether the congestion is caused by statistical fluctuations or enemy action.

2.3.2.1 Static Routing

Although not yet definitive, significant progress has been made on routing and related issues during the last two years. A recent multicommodity routing algorithm due to Cantor and Gerla† is especially significant from a theoretical point of view, in that it provides a mathematically sound procedure for solving static problems with a computational efficiency comparable to the best available heuristic methods. In particular, the Cantor-Gerla algorithm can be used to find routes that minimize an approximation to the average delay on a packet-switched network. Part of our work during the past year has been devoted to devising and programming an extremely efficient variant of this algorithm.

---

†Cantor, D.G. and M. Gerla, "Optimal Routing in a Packet Switched Computer Network", IEEE Trans. on Computers, Vol. C-23, Oct. 1974.

2.3.2.2 Quasistatic Routing

In addition, we have developed a new quasistatic algorithm in which the computation is uniformly distributed over the nodes of the network. Each node successively calculates its own routing tables based on successive updating information from neighboring nodes. It has been proven that if the network inputs are stationary and the network is not changing, and if certain other requirements are satisfied, then the average delay converges to the minimum delay over all choices of fixed routing tables. In other words, under static conditions, the algorithm solves the same problem as static routing algorithms such as the Cantor-Gerla algorithm, whereas the ARPANET algorithm does not.[†]

2.3.2.3 Dynamic Routing

A good deal of our effort has also been directed towards dynamic routing. A new model has been developed for this problem in which the contents of the queues at the nodes are viewed as continuous variables, rather than as an integer number of messages or bits. This macroscopic point of view not only provides a model that is analytically simpler than the classic finite-state models, but also agrees with the fact that the effect of any single message on the total system performance is minimal.

Two approaches to dynamic routing have been followed, one of which is aimed at finding the optimum closed-loop feedback control solution with its attendant advantages (principally, a reduction of sensitivity to perturbations in the system parameters and inputs). Mathematically this problem can be formulated as a linear optimal control problem with state and control variable linear inequality constraints. Using this approach, we have investigated how to minimize the average message delay while disposing of whatever backlogs may exist in the network at any particular point in time.

[†]The principal distinction is that the ARPANET strives for what is called "user optimization",rather than for "system optimization". (See Section 3.4.3)

A comprehensive feedback solution has been obtained for the case in which all backlogs have the same destination.

The second approach involves a shift in objectives: rather than trying to minimize the average delay in emptying a network of backlogs, we seek instead to minimize the maximum time required to do so. A variety of different multicommodity flow problems can be formulated along these lines. A computer program for solving some of these problems works well for small networks, but our current implementation is unable to acommodate networks with more than 18 or so links.

2.3.3 <u>Network Topology</u>

The dependence of reliability on network topology is one of the most difficult areas we have addressed so far. The fundamental question one would like to answer asks what system architecture will yield networks whose reliability increases as fast as possible as a function of network size (more generally, of its cost or complexity). Here "reliability" means the probability that the network will meet meaningful performance criteria subject to reasonable assumptions about node, link, and traffic statistics.

This problem becomes complicated at a very elementary level. We have found it difficult to devise useful measures of how performance or complexity vary as a function of topology even when node and link parameters are constant and a fixed routing pattern that optimizes the expected traffic flow is used. The difficulty is compounded many-fold when it is realized that with adaptive routing--which any military data network must certainly have--a single network topology corresponds to a large class of "virtual topologies", one for each possible routing policy. Although we have little progress to report to date, we anticipate that new insights should develop as we gain new understanding of how performance varies with routing policy when the topology is held fixed. We are specifically interested in how to structure the topology into subnets, so that (1) routing complexity grows only moderately as new subnets are added, while (2) still retaining most of the potential gain in reliability which greater network size should make possible.

2.4 Communication Links

In our taxonomy a network depends upon its communication links for the

- dissemination of protocol data, the
- control of transmission errors, and the
- estimation of network parameters.

All three activities affect reliability directly, but in very different ways.

2.4.1 Protocol Data

In a message switched or packet switched network, one normally precedes each message or packet with a header and follows it with a trailer. Various fields in the header and trailer contain all the relevant information necessary for control of the message, such as parsing information for successive messages on a link, source and destination information, sequencing information for messages on any given virtual path, and error control information on each link. In addition there is a need for a variety of other kinds of control information such as routing update information, flow control information, network status information, information to set up new virtual paths, and so forth. Clearly, care must be taken that strategies aimed at improving transmission efficiency are not self-defeating (or worse) due to increases in the protocol data required to implement them.

In order to investigate the efficiency with which links in a network are used, emphasis must be placed on the words "protocol required". It is necessary to ask what protocol information must be carried by a link, rather than taking it as an inviolate principle that every message or packet must be transmitted as a unit with all of its control information in fixed fields. With this broader viewpoint one can separately investigate a number of questions about the individual links of a network, for example, how to encode the length of each message on a link and how to encode the fact that a message is starting. We have investigated this question both in relation to how many bits are required by such encodings and to what kind of queues are generated on a link by such encodings.

2.4.2 Error Control

We have also studied the efficiency of transmission error control procedures on a link. Although current line control procedures turn out to be quite efficient in this regard, it has proved to be conceptually interesting to investigate optimal performance. Specifically, the two-way protocols involved here provide insight into how to investigate deadlocks for more complex protocols.

2.4.3 Estimation

Finally, it is important to realize that the parameters needed for proper operation of the network and implementation of various algorithms are usually not available directly. It is therefore of major importance to design good procedures for estimating these parameters. Specifically, the derivative of the total delay per unit time on a link, with respect to the flow rate over that link, is of major importance in routing problems. We have therefore concentrated on devising procedures for the robust estimation of these derivatives.

3. Current Status and Prospective Work

In this section we discuss in some detail the work that has been accomplished during the past year, and indicate that which we anticipate undertaking during the year to come. The discussion follows the taxonomy introduced in Section 2.

3.1 Computer Subnet

3.1.1 File Allocation

The modeling and analysis of problems of dynamic file allocation have been investigated in [1]-[4], in which a finite state Markov process is introduced to represent the current situation in the network. Each state component corresponds to one computer in the network and indicates whether the computer is presently operating or is out of order, and if the former is true, whether it presently carries a copy of the file or not. The model has been designed to incorporate the following parameters:

- Number of copies. The optimal number of copies of files and their location depends on the operating cost, topology of the network and future expected changes.
- Node failures. A strategy is designed to minimize the probability that the system will lose all copies of the information files because of computer failures. This has been done by introducing an additional term in the cost function representing a high cost if all copies of the file are lost.
- Query and updating traffic. In addition to requests for use, a significant amount of communication traffic is required for updating the contents of the files. Clearly, the system becomes more reliable as more copies are kept, but storage costs and updating traffic requirements tend to decrease the optimum number of copies.

- Parameter estimation. The model assumes knowledge of various operational parameters such as request rates, probabilities of failure and recovery[†], etc. Methods of estimating these parameters on line and incorporating the estimates into the decision procedure are given in [1].

Using the Markovian model and a cost criterion determined by the operating cost of the network (costs for storage, transmission and for losing all copies of a file), dynamic programming has been used to obtain the optimal decision strategy (i.e., the optimum policy of writing and erasing copies of files) as a function of the state of the network.

The effect of various parameters has been studied in several numerical examples where the model has been used [4]. We find that savings of up to 50% of the operation cost can be obtained by using a dynamic instead of a static allocation strategy. It has also been observed [4] that the failure probabilities affect the optimal strategy much more strongly than do the recovery probabilities. For a three computer network with specific parameters (request probabilities = 0.4, 0.6, 0.8, transmission cost = 1, storage cost = 0.25, cost of losing all copies = 1000), it has been shown [4, p.133] that if the probability of failure p_f is 1% or higher, copies must be kept in all working computers. For p_f = .1% and p_f = .01%, keeping only two copies in the system is sufficient. Clearly, the inclusion of updating traffic will tend to increase the operation cost and decrease the optimal number of copies of the file to be kept in the system. This effect is illustrated in Figure 1 for a three-computer network , in which ρ represents the ratio of updating traffic rate to query traffic rate, C_s is the storage cost and the transmission cost has been normalized to unity. In region 111 the optimal decision is to keep copies in all computers, in

---

[†]Recovery probability is the probability that a computer is restored at time t given that it was out of order one time unit earlier. It is assumed that a failure destroys all information in the computer.

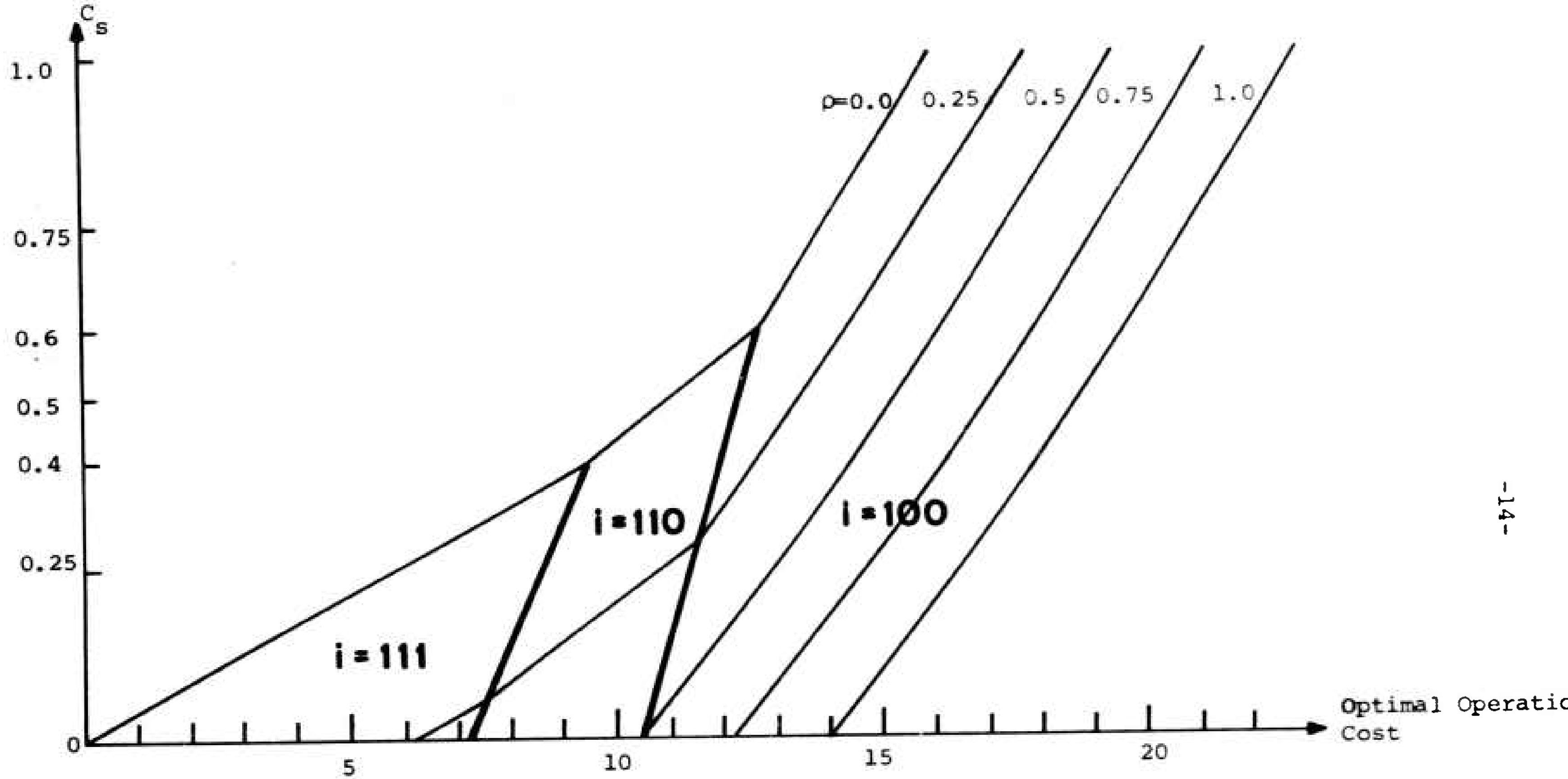


Fig. 1 Curv of total minimum cost vs. storage cost with ρ as a parameter.

region 110 copies should be kept only in computers 1 and 2, and in region 100 a copy is kept in computer 1 only.

3.1.1.1 Future Work

More work has to be done in investigating the model that has been designed in [1]-[4]. One problem is that being a finite state Markov model, the number of states increases exponentially with the size of the network. Promising first steps in the direction of circumventing this difficulty have been taken in [4], where it was shown that for a network with symmetric parameters, the model simplifies considerably.

Another important direction of research in this area is to investigate schemes for distributed control of dynamic file allocation. The model described above assumes complete knowledge of the network state by a central controller. As always in a computer network, such a strategy will require a large part of the communication facility to be devoted to status and protocol transmissions. It is therefore important to investigate attractive decentralized strategies. First steps in this direction have been taken in [2].

3.1.2 Load Sharing

One of the reasons why data communication networks are important is that in a large system involving many computers, overall reliability may be increased by load sharing. The advantages to be gained may be studied quantitatively by modeling both the computers and the communication channels as queues, and evaluating the steady-state expected time to process a job [5 , 6].

Bounds: Consider for simplicity a set of identical computers, at each of which jobs arrive independently with exponential interarrival and service times. For the ith computer, the mean delay from job arrival to job completion is

$$\frac{1}{\nu-\lambda_i} \quad , \qquad 0 < \lambda_i < \nu$$

where $1/\nu$ is the mean service time and λ_i is the mean number of jobs arriving per second. Under these circumstances, it is easy to show that the mean delay averaged over a system of N computers is minimized when each is assigned (on the average) an equal fraction of the total number of jobs. This basic idea of balancing the average load may be extended to more general situations, involving unequal computer speeds and the communication delays incurred in transmitting job requests and returning completed job results.

We call these generalized techniques for reducing mean job delay "Statistical Load Sharing", in that decisions on sending "what job where" are made purely on the basis of averages, and not at all on the basis of the system state (i.e., on how many jobs are "queued where when".) Thus the minimum average delay obtainable by statistical load sharing is an upper bound on the true minimum, which could in principle be obtained with dynamic (state-dependent) system control. It is interesting, however, that in the case of identical computers and fast enough communication circuits, the maximum system throughput obtainable (at infinite delay) by dynamic and statistical load sharing are the same, a result that follows by comparison with the idealized performance that would be possible if all computers were co-located.

Multicommodity Flow Formulation: Questions of load sharing have been considered before [7 , 8], and where the work discussed here overlaps previous studies the results substantially agree. An important new result we have obtained is identification of statistical load sharing as a multicommodity flow problem which is amenable to efficient solution by means of the Cantor-Gerla algorithm [9].

The transformation leading to this identification is best presented via a small example. Consider the three-computer network shown in Figure 2a. The corresponding topological representation as a multicommodity flow problem is shown in Figure 2b, in which each computer is associated with

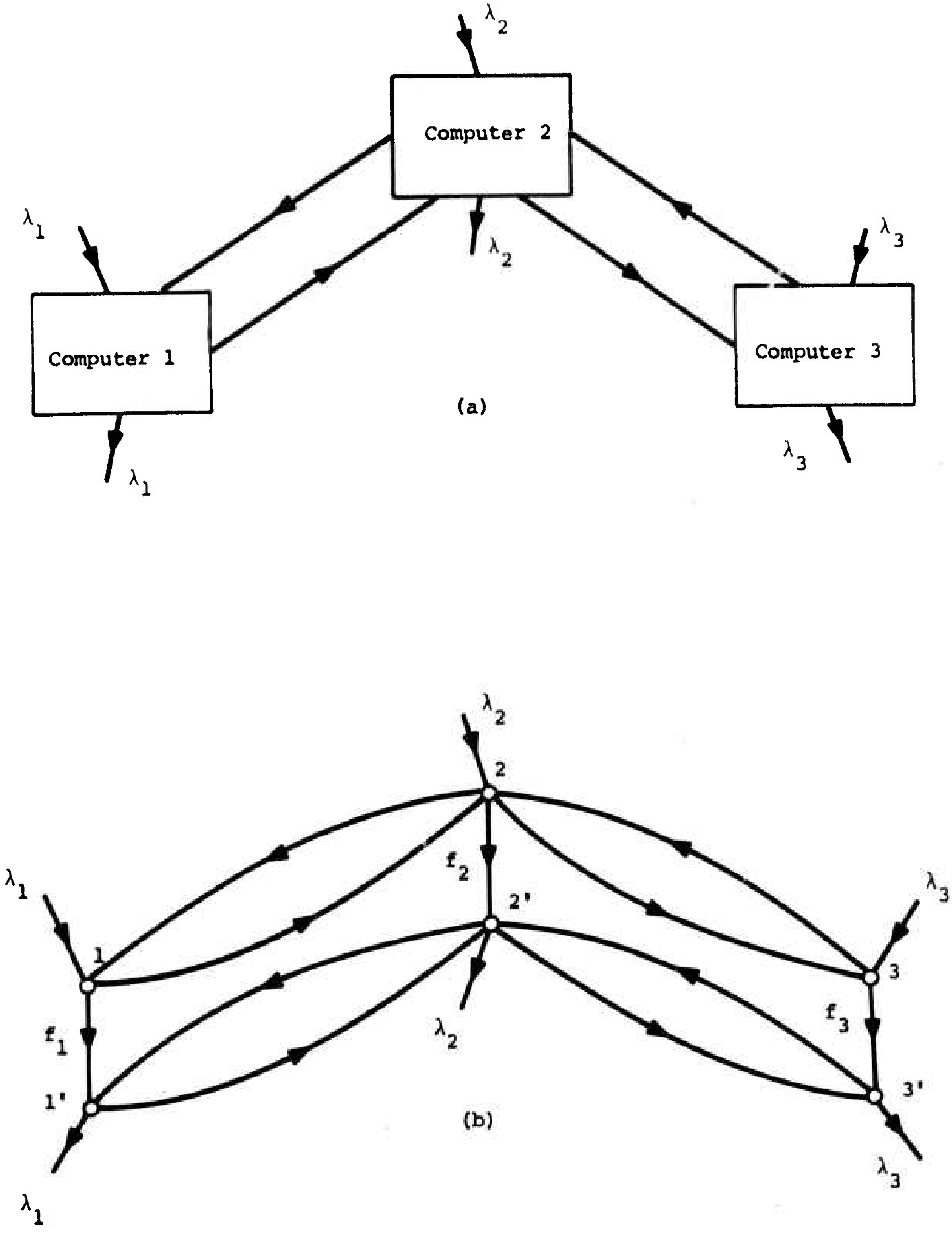


Figure 2
(a) Three computer system with a partially connected communication network.
(b) Multicommodity flow formulation for the network of (a).

an input and an output node, and each physical communications channel is associated with two topological links (one for programs and one for results). The success of the representation rests upon the fact that in an optimum solution it never happens that any computer transmits both programs and results: either it helps others, or asks for help, but not both.

Computer/Communication Interaction: Although a computerized algorithm (such as Cantor and Gerla's) must be used to solve arbitrary statistical load-sharing problems, a great deal of insight may be gained by focusing on the general nature of the solution. Three characteristics of the interaction between the computer system and the communication system seem especially important [5]:

- There is a minimum total system load threshold beneath which job-sharing is undesirable (the savings due to load equalization do not make up for the losses due to communication delays).
- There is a maximum probability of sending a job elsewhere, which will be reached asymptotically as the total system load increases provided the computer system saturates before the communication system does. This asymptote is just that probability which achieves balanced computer loading.
- If the communication system saturates before the computer system does, the asymptotic probability will not be reached. If the communications network is bad enough, the job sharing threshold will never be reached.

3.1.2.1 Future Work

Additional work on statistical load sharing, with the computers modeled more realisticaly than by a single server queue with exponential service times, would be possible. The study of dynamic load sharing, however, seems more interesting. Specifically, having established the relationship between load sharing and routing when both are viewed statistically, it is natural to ask whether a similar relationship exists in the dynamic case. The problem will be somewhat different here than in our dynamic routing work, since each of the three operations (job request, job calculation, and return of the answer) must be accounted for explicitly.

3.2 Computer/Communication Interface

As mentioned in Section 2.2, we have not yet progressed much beyond background work in areas such as flow control, congestion, and topology which are tied especially intimately to problems of routing. These and the related topics of recovery from congestion and deadlocks will be major topics of investigation in the coming year. Prospective work in flow control is discussed here, while recovery and topology are treated in Subsection 3.3.

3.2.1 Future Work

There has been very little progress of a general systematic kind on flow control, although there has been considerable work on specific strategies. One type of strategy, as exemplified by the ARPANET [10] and generalized in various window strategies [11], exerts end-to-end flow control by limiting the number of messages or packets which can be outstanding in the network on a given virtual path. Another type of strategy, exemplified by the NPL network [12], limits the total number of messages or packets outstanding in the network. The end-to-end strategies have the advantage of limiting input most stringently on those virtual paths using the most congested links. They have the disadvantage that congestion can still occur if too many virtual paths are active at once and use too many links in common. The NPL strategy has the disadvantage of being insensitive to the location of potential congestion and also the disadvantage that the permits used to limit the number of messages might not be where they are needed.

Two questions that we shall attempt to answer in this area are the following: 1) given that an end-to-end flow control strategy is to be used (i.e., a strategy in which permission to enter the network on a given virtual path is a function solely of previous delays on that path), what strategy optimizes the tradeoff between rejecting messages and preventing congestion. Obviously the answer here will depend upon the network model, but if an answer can be given for a sufficiently simple model, it should enhance our understanding of end-to-end flow control. 2) Is there any sensible way to use information about network queue lengths at the input ports to obtain substantially better flow control than with end-to-end procedures? This question can be approached by first ignoring the cost of getting the information at the input ports, but later taking that cost into account.

Along with these specific questions, we would like to develop a general framework within which to study flow control. Some progress along these lines has been made by Chou and Gerla [13], but their interest was in systematization for the purpose of developing general simulation software rather than conceptual systematization.

3.3 Communication Subsystem

3.3.1 Recovery

Regardless of the effectiveness of adaptive routing techniques, it is statistically inevitable that congestion will occur. Thus it is essential to understand the problem of deadlocks in networks and their relationship to congestion and timeouts.

3.3.1.1 Future Work

We can roughly distinguish two different kinds of timeouts--asynchronous timeouts where the time inverval chosen affects performance but can not cause deadlocks, and critical timeouts where if the timeout occurs before or after some other event either a deadlock will occur or some event will occur which requires another critical timeout for recovery. In data networks the nodes generally operate asynchronously with respect to each other and, due to error control, the time required to transmit a frame on a link is subject to uncertain delays. In such situations, critical timeouts are likely to cause subtle deadlocks which arise only in abnormal conditions. Thus we want to determine under what conditions congestion recovery can be accomplished without any critical timeouts. Merlin and Farber [14] have shown that critical timeouts are necessary if messages are allowed to disappear without any trace within the network. Such message disappearances are certainly possible if a node malfunctions in a certain way, but we are not convinced that congestion or even complete link or node failure need cause messages to disappear with no indication of the fact remaining.

Part of our effort in the area of deadlocks will be devoted to trying to develop an appropriate set of tools. There are a number of tools available in computer science such as Petri nets, multi-process synchronization primitives, and deadlock prevention algorithms, but none seems entirely appropriate for data communication networks.

3.3.2 Routing

Routing algorithms can be classified according to their adaptivity. At the two ends of the scale lie completely static and completely dynamic strategies. In between is quasi-static routing, where changes of routes are allowed only at time invervals that are large compared to the dynamics of the system.

3.3.2.1 Static Routing

We have already remarked upon the fundamental position held by the Cantor-Gerla algorithm in our current understanding of optimum multicommodity flow problems. Specifically, the algorithm finds [9] a static flow vector $\vec{f}$ which simultaneously

(a) satisfies a requirements matrix $[r_i^j]$ whose components are steady flows from node i to node j,

(b) has link components f_ℓ that are all non-negative and no greater than the capacity c_ℓ of the corresponding link in the network, and

(c) minimizes the objective function

$$\sum_{\text{all links}} \frac{f_\ell}{c_\ell - f_\ell}$$

The motivation for this objective function is that under certain assumptions its value approximates the mean delay in a packet switched network [15]. It is worth noting, however, that the mathematical problem formulation involves steady flow requirements rather than packets, and that the solution for each commodity is therefore "filamentary" in nature, by which we mean a lattice of flows with the property that flow is conserved at every intermediate node as shown in Figure 3. In current practice, the proportional division at a node of the flow of a commodity in the filamentary solution would be interpreted as the relative frequency with which packets of that commodity should be routed from the node via the corresponding links, and buffers would be used at each internal node to store the packets enroute.

Part of our work during the past year has been devoted to implementing the Cantor-Gerla algorithm efficiently. In the process of doing so, two significant advances have been made in computational efficiency. The first [16] concerns two improved algorithms for determining the set of shortest distances between all pairs of nodes in a sparse network, to each link of which has been assigned a non-zero length.† The improvements obtainable with these algorithms depends upon the topology and size of the network; for the most advantageous case (a tree network) the order of growth of computational complexity is reduced from N^3 (for brute-force calculation) to N^2, where N is the number of nodes.

The second advance in efficiency relates to decomposing and representing flow solutions in such a way that not only the vector $\vec{f}$, but also the routing of each individual commodity, can be rapidly calculated. Preliminary results [17] indicate that significant advantages may be obtained in cases of practical interest by decomposing the flow into trees (one for

†After one of these new algorithms had been implemented and documented for publication, it was discovered that substantially the same algorithm had been worked out by the Network Analysis Corporation and disclosed in its Fourth Semi-Annual Technical Report to ARPA, dated 15 Dec. 1971, under Contract No. DAHC 15-70-C-0120.

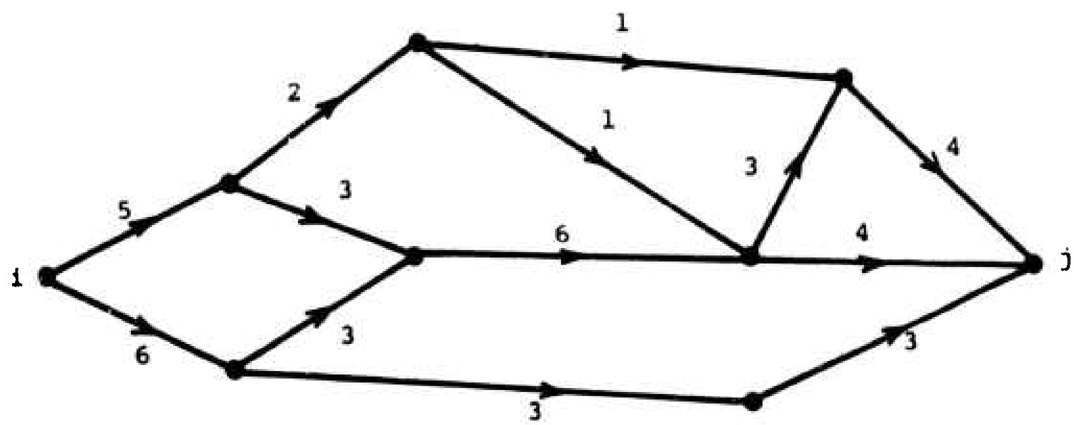


Figure 3

A lattice of filamentary flows carrying 11 units per second from node i to node j. Flow is conserved at every intermediate node, so that the flow of commodity i → j across every cutset is also 11. The complete network flow solution is the superposition of many such lattices, one for each source-sink pair.

each destination) rather than into extremal flow, as in the original Cantor-Gerla algorithm.

3.3.2.2 Quasistatic Routing

Although our distributed algorithm [18] has not yet been analyzed for the case of slowly varying input statistics and changing network topology, a local computation algorithm has a number of heuristic advantages over a centralized Cantor-Gerla algorithm for such circumstances. First, local changes can be responded to quickly at local nodes since the relevant information is present there. Second, the time varying statistics needed by the algorithm can be calculated dynamically at the local nodes as a natural part of the algorithm rather than requiring separate protocols for transmitting this information to a central node. Finally, there is no handover problem, as there would be were the central node to malfunction.

One of the most interesting features of the new algorithm is that the routing tables for each destination in the network are guaranteed to be loop free at each iteration of the algorithm. The loop free property was designed into the algorithm for two reasons--first to reduce the expected delay at each step of the algorithm, and second, to prevent a potential deadlock in the protocol for transmitting updating information in the algorithm. It appears that this loop-freedom property might also be important in the design of deadlock free higher level network protocols.

The new algorithm is quite similar to the routing algorithm used in the ARPANET. One difference, essential for convergence to minimum expected delay, is that the updating information consists of differential delays rather than absolute delays. Another difference is that the updating information is sent with a certain ordering in time; this both prevents loops and increases the speed at which information propagates through the network.

3.3.2.3 Dynamic Routing

For purposes of routing analysis, we consider a network to consist of nodes and links, which are basically message storage-and-switching areas and data communication lines, respectively. In addition to storage, each node is responsible for the proper dispatching of all messages entering that node either from users or from adjacent nodes. Let the collection of nodes be denoted by $\mathcal{N} = \{1,2,\ldots, N\}$. For every $i \in \mathcal{N}$, denote

$D(i)$ = collection of lines outgoing from node i

$I(i)$ = collection of lines incoming to node i

We also denote by $\mathcal{L}$ the collection of all links in the network (all lines are taken to be simplex, that is, to transmit data in one direction only):

$$\mathcal{L} = \{(i,k), \text{ such that } i, k \in \mathcal{N} \text{ and there is a direct link connecting } i \text{ to } k\}$$

<u>Dynamic Routing Model</u>: In formulating the problem of completely dynamic control of message routing [19], we imagine that at each node $i \in \mathcal{N}$ we have N-1 buffers in which at every time t we store all messages, packets, bits, etc. whose destination node is $1,2,\ldots, i-1, i+1,\ldots N$ respectively, disregarding the node of origin. The number of messages of each type at any time t may be measured in arbitrary units, but we assume that the units are such that after appropriate normalization, the contents of the buffers can be approximated by continuous variables. We let

$x_i^j(t) \triangleq$ amount of message traffic at node i at time t whose destination is node j.

The <u>inputs</u> are

$r_i^j(t) \triangleq$ rate of message traffic with destination j arriving at node i from users of the network

Finally, we define the control:

$u_{ik}^{j}(t) \triangleq$ portion of the capacity of link (i,k) used at time t for traffic with destination j.

Having characterized the message flow elements as above, we may write the time rate of change of a given state $x_i^j(t)$ as

$$\dot{x}_i^j(t) = r_i^j(t) - \sum_{k \in D(i)} u_{ik}^j(t) + \sum_{\substack{\ell \in I(i) \\ \ell \neq j}} u_{\ell i}^j(t)$$

$$i \in \mathcal{N},\ j \in \mathcal{N},\ j \neq i$$

The positivity constraints

$$x_i^j(t) \geq 0 \quad ; \quad u_{ik}^j(t) \geq 0$$

and the capacity constraints

$$\sum_{j \neq i} u_{ik}^j(t) \leq c_{ik} \quad , \quad (i,k) \in \mathcal{L}$$

must be satisfied for all t, where

$c_{ik} \triangleq$ capacity of (i,k) in units of traffic/unit time, where $(i,k) \in \mathcal{L}$

Dynamic Optimization: The integral

$$D = \int_{t_o}^{t_f} \left[\sum_{i,j} x_i^j(t) \right] dt$$

gives the total delay experienced by all the messages in the network over the time interval $[t_o, t_f]$. Hence, we may state the dynamic routing problem for the minimization of total delay as the following optimal control problem [19]:

> At every time t, given the knowledge of the network congestion $\{x_i^j(t);\ i, j \in \mathcal{N},\ j \neq i\}$, dynamically determine all $u_{ik}^j(t)$ so as to minimize the total delay D.

The fundamental approach we have taken towards solving this problem is as follows: We begin by developing the necessary conditions associated with Pontryagin's minimum principle. Fortunately, these conditions are also sufficient, which greatly enhances their utility. In the spirit of dynamic programming, we use these necessary conditions to construct a set of optimal trajectories backward in time which parametrically span the entire state space with optimal trajectories and their associated controls. In so doing, we construct the optimal feedback routing algorithm. The entire approach exploits very heavily the linearity of the model.

Using this technique, we have investigated the problem of disposing of backlogs that may exist in the network at any particular point in time. A comprehensive feedback solution when all backlogs go to a single destination and the inputs are constant in time is given in [20]. The nature of the solution is that the state space is divided into conical regions bounded by hyperplanes, and the optimal routing control is constant within each of these regions. At the time some of the initial backlogs go to zero, the state goes into a new region and a new control value is used. This procedure continues until all buffers are emptied.

Another approach to the solution of the dynamic optimization problem involves replacing the nonnegativity constraints on the states by penalty functions. This approach can incorporate arbitrary (known) inputs, but provides a nonfeedback solution. Introducing penalty functions also results in several new theoretical difficulties, especially singular controls, which may however be resolvable by exploiting the linearity of the problem. For example, we have observed, at least for traffic with a single destination, that the singular controls are constant points on the surface of the constrained region. By distorting the constrained region and approximately solving the resulting problem we can avoid the computational difficulty that singular controls usually cause.

Generally, the numerical solution of a dynamic optimization problem requires many integrations of differential equations as well as the solution of a static optimization problem at every time step (the minimum principle). The linearity of this problem, however, allows us to solve the differential equations analytically rather than numerically.

In addition, since the control also appears linearly, the minimum principle leads to a linear programming problem, for which there are numerous efficient algorithms. Finally, by means of parametric linear programming techniques, the linear programming algorithm ne d be invoked only when the control law must be changed, and not at every time instant. These techniques are expected to save considerable computer time. At present, we have set up the problem and are in the process of programming the algorithm.

Up to now, all examples considered have included single destination traffic. It is anticipated that certain difficulties may arise in the general case. First, it may no longer be true that singular controls are fixed points in control space. Second, it may become more difficult to perform the proper boundary distortion to sidestep singularity problems.

3.3.2.4 Minimax Routing

A second approach [21] to the problem of dynamic routing involves a change in basic objective while keeping the same model: rather than trying to minimize the average delay, we focus instead on minimizing the maximum delay. A wide variety of related network models and performance criteria can be approached from this point of view. The simplest problem with which to start, however, is the steady multicommodity flow case (which was also our viewpoint in discussing the Cantor-Gerla algorithm).

Steady Input Flows: For this case we again have a requirements matrix $[r_i^j]$ whose elements are the flows from node i destined for node j $(1 \leq i, j \leq N)$, where each such flow is a separate commodity. Our objective is to route the commodities thru a network--defined by a link capacity matrix $[c_{\ell k}]$--in such a way that the greatest "saturation level" among the links is as small as possible. By saturation level we mean the ratio of the total flow in a link to the link's capacity, so that our objective is to determine the routing policy which minimizes

$$\max \frac{f_{\ell k}}{c_{\ell k}} \qquad (\ell, k) \varepsilon \mathcal{L}$$

An alternative, equivalent way of stating the problem is to consider a network $[\alpha \, c_{\ell k}]$, where $\alpha > 0$ is a scaling factor, and ask for the minimum value of α for which a feasible flow exists.

It is easy to see on physical grounds that this problem has a unique solution: as α reduces to some value, say α_1, a subset of the links is bound to saturate, so that the requirements matrix can not be satisfied for $\alpha < \alpha_1$. Although less obvious, it is not difficult to prove that for given $[r_i^j]$ and $[c_{\ell k}]$ there is a unique minimal subset (say S_1) of links that saturate at $\alpha = \alpha_1$, and a unique subset (say R_1) of commodities that must flow thru S_1. Thus shrinking the network produces the unique triple $(\alpha_1 \quad S_1 \quad R_1)$ with the property that no feasible flow exists if either

(a) any link (ℓ,k) in S_1 has its capacity reduced beneath $\alpha_1 c_{\ell k}$, or if

(b) any requirement r_i^j in R_1 is increased while all other commodity requirements are held fixed.

Although we can not reduce further the capacities in saturation set S_1, the remaining links in the network $[\alpha_1 \; c_{\ell k}]$ need not be saturated. We may therefore next seek to minimize the parameter α in the network obtained by clamping the capacities in S_1 and scaling the rest of the network, so that

$$c_{\ell k} \leftarrow \begin{cases} \alpha_1 \; c_{\ell k} & (\ell,k) \in S_1 \\ \alpha \;\; c_{\ell k} & \text{otherwise} \end{cases}$$

Let α_2 be the minimum value of α for which a feasible flow now exists. The implication is that a second set of links (say S_2) must saturate, and that an additional† set (say R_2) of commodities must flow thru S_2. As before, it can be shown that the triple $(\alpha_2 \quad S_2 \quad R_2)$ is also unique.

The procedure may now be iterated by clamping the links in S_2 at capacity $(\alpha_2 \; c_{\ell k})$, and scaling those links of the network that are neither in S_1 nor S_2. Continuing to iterate in this way, we ultimately generate a finite number of triples

$$\begin{array}{lll} (\alpha_1 & S_1 & R_1) \\ (\alpha_2 & S_2 & R_2) \\ & \vdots & \\ (\alpha_M & S_M & R_M) \end{array}$$

---

†Some of the commodities in R_1 may also utilize S_2 (they must get from their sources to S_1, and from S_1 to their destinations, somehow).

where S_m and S_n, and R_m and R_n, are disjoint for all m and $n \neq m$, and $\alpha_1 > \alpha_2 \ldots > \alpha_M$. The network defined by links (ℓ,k) with capacities

$$c^*_{\ell k} \triangleq \alpha_m c_{\ell k} \qquad (\ell,k) \in S_m, \qquad m = 1,2,\ldots, M$$

may be interpreted as the "smallest" network obtainable from $[c_{\ell k}]$ by scaling, which still satisfies $[r_i^j]$. Commodities in any set R_m will flow (in toto) thru links in S_m, may flow thru links in saturation sets of index $n > m$, and will not flow at all thru saturation sets of index $n < m$.

Any feasible flow $\vec{f}$ for $[c^*_{\ell k}]$ that satisfies $[r_i^j]$ is, of course, a† feasible flow for the original network $[c_{\ell k}]$, and the numbers α_m $(m=1,2,\ldots,M)$ are the saturation levels produced in the link sets S_m when $\vec{f}$ flows in $[c_{\ell k}]$. As in the case of the Cantor-Gerla algorithm, the flow solution $\vec{f}$ is steady and filamentary.

Deterministic Backlogs: The problem of emptying a network of an accumulated backlog of traffic as fast as possible is mathematically similar to the steady flow case. Let the backlogs be denoted by a matrix $[x_i^j]$, and consider the flow matrix obtained by dividing each x_i^j by a parameter τ (whose dimension is seconds). If τ_1 is the smallest value of τ for which a feasible flow exists over $[c_{\ell k}]$, then τ_1 is the minimum time necessary to clear the network of all backlogged traffic. Although different in their dimensionality, τ_1 and α_1 are numerically equal.

Having solved for τ_1, we can convert the backlogs to steady flows by defining

$$[r_i^j] \triangleq \left[\frac{x_i^j}{\tau_1}\right]$$

and carry out the iterative solution of the triples $(\alpha_m\ S_m,\ R_m)$ for $m = 2,\ldots M$ exactly as before. Any flow $\vec{f}$ that satisfies $[r_i^j]$ and is feasible for the resulting minimal network $[c^*_{\ell k}]$ will empty the backlogs in τ_1

---

†We assume here that $\alpha_1 \leq 1$.

seconds, and use minimal fractions $\alpha_1 = 1 > \alpha_2 > \ldots \alpha_M$ of the capacities $[c_{\ell k}]$ while doing so. The steady-flow nature of the resulting solution (over the time inverval $0 - \tau_1$) is indicated in Figure 4a.

Although the time τ_1 required to empty the backlogs x_i^j corresponding to the commodity set R_1 can not be reduced, it is possible to modify the solution of Figure 4a by compacting the flows so that the link sets S_2, $S_3, \ldots, S_M$ are saturated over progressively shorter invervals $\tau_2, \tau_3, \ldots, \tau_M$ as shown in Figure 4b. The flow transformation is straightforward: Let $|S_2^*|$ denote the total residual capacity of S_2 after subtracting the flow thru S_2 of commodities in R_1. Then

$$\tau_2 = \frac{|X_2|}{|S_2^*|} \quad \text{seconds}$$

where $|X_2|$ denotes the sum of all commodities in R_2. Similarly, we can compact the flow of commodities in R_3 and saturate S_3 for

$$\tau_3 = \frac{|X_3|}{|S_3^*|} \quad \text{seconds}$$

where $|S_3^*|$ is the residual capacity of S_3 after subtracting the compacted flow thru S_3 due to commodities in R_1 or R_2.

The general applicability of the above procedure is guaranteed by the decreasing nature of the sequence $\alpha_1, \alpha_2, \ldots, \alpha_M$. The only links that will remain always unsaturated are those belonging to sets S_m for which R_m is empty. Moreover, although the flow solution is now only piecewise constant

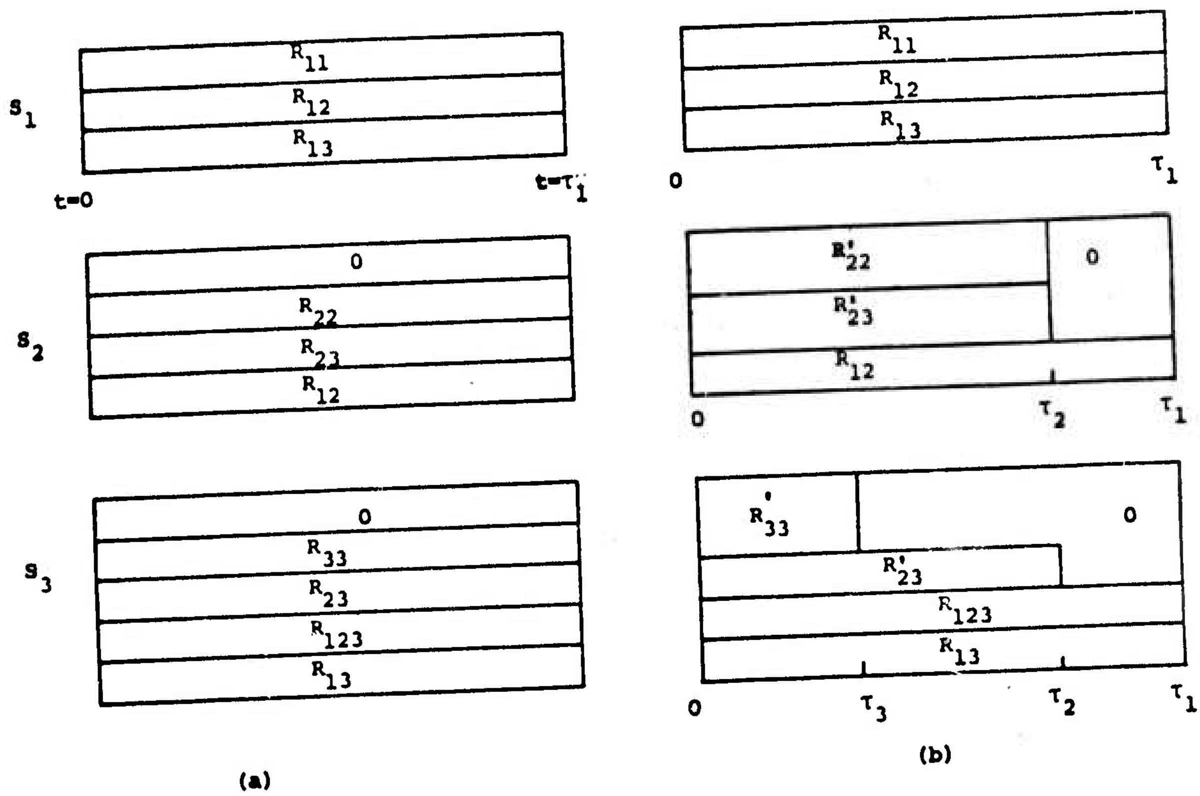


Figure 4 Steady (a) and Compacted (b) Transient Flows

Here the notation R_{mm} means commodities in R_m that flow only in link saturation set S_m. Similarly, R_{mn} means those commodities in R_m that flow also in link saturation set S_n, $n > m$.

R_{123} means those commodities in R_1 that flow also in both S_2 and S_3.

The symbol 0 denotes regions of unused link capacity.

in time, it is still true that no internal buffering is necessary; this follows from the fact that the flow-compaction can always be done proportionally, so that continuity of flow at each intermediate node is maintained for every commodity.

Despite the attractive features of the flow solution obtained in this way, it is optimum (in a minimax sense) only over the subset of feasible flows with the continuous-flow property. The simple examples in Figure 5 show that it is sometimes possible for $m > 1$ both to deliver commodities in R_m and to terminate the flow in S_m earlier when internal buffering is allowed.

Backlogs Plus Steady Inputs: The situation in which backlogs are to be emptied as soon as possible, while simultaneously accommodating steady inputs, involves only a minor conceptual extension. Specifically, in order to determine the minimum time to empty, we consider the requirements matrix

$$[r_i^j] + \frac{x_i^j}{\tau}]$$

where the r_i^j denote the steady flows and the x_i^j the backlogs. We then solve for the smallest value of τ, say τ_1, such that a feasible flow for the network $[c_{\ell k}]$ exists. Having determined the triple $(\tau_1 \;\; S_1 \;\; R_1)$, we then continue the iteration as before, using $[r_i^j + x_i^j/\tau_1]$ as the requirements matrix and clamping successive link saturation sets S_m at their minimum allowable capacity by the substitution

$$c_{\ell k} \leftarrow \alpha_m \, c_{\ell k} \qquad \text{for } (\ell,k) \in S_m$$

The final flow vector $\vec{f}$ satisfies requirements $[r_i^j + x_i^j/\tau_1]$ and is feasible for the minimal network $[c^*_{\ell k}]$. Transient flow components may again be compacted as in Figure 5.

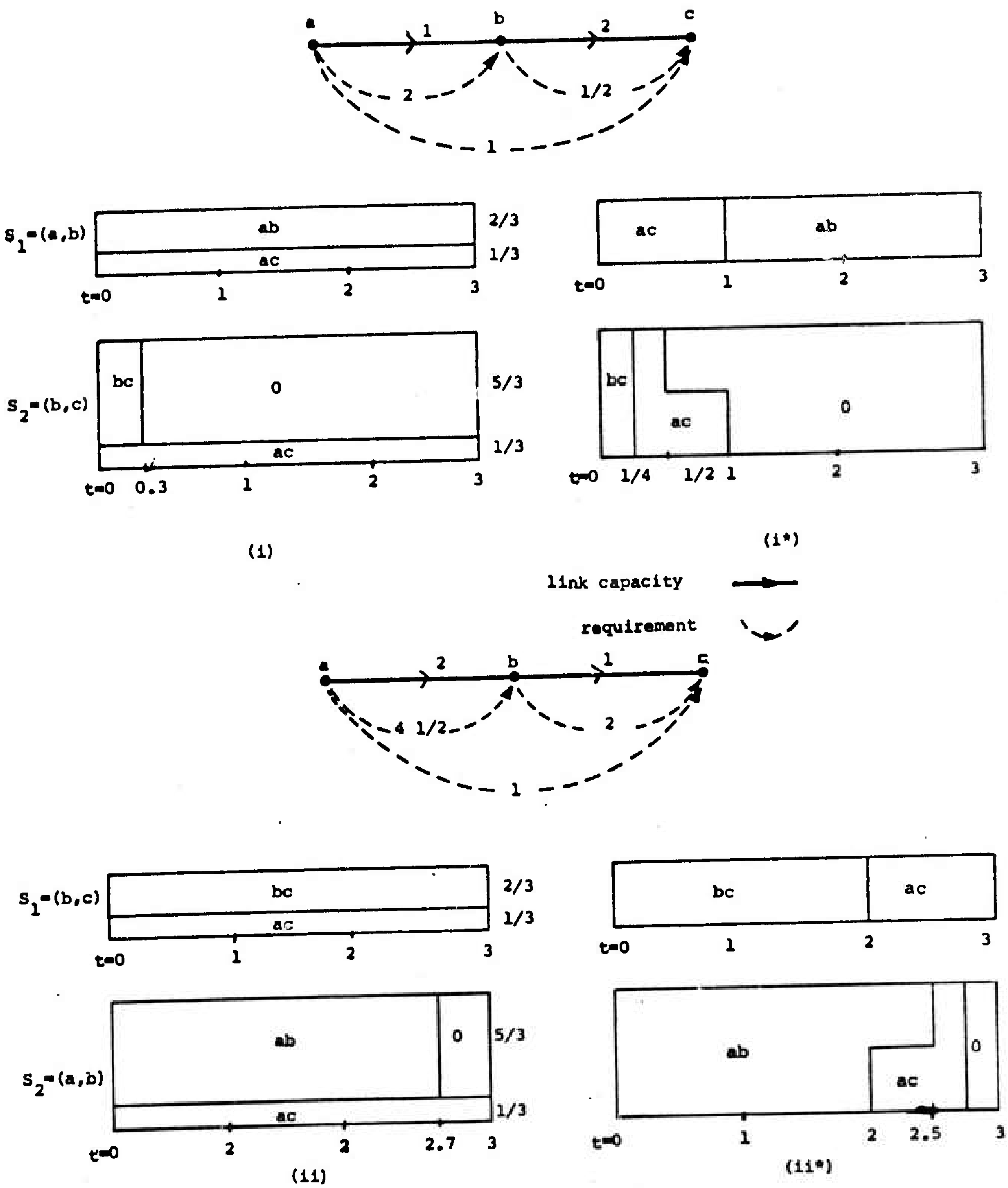


Figure 5 Piecewise-Constant (i)(ii) and Maximally Compacted (i*)(ii*) Flows

<u>Random Inputs</u>: From the point of view of network reliability, primary interest attaches to the case where inputs are random. Indeed, otherwise there would be no way for (finite) backlogs to occur.

The key to extending the minimax formulation to random inputs lies in a result of queueing theory. Consider a single-server queue in which the intervals between customer arrivals are exponentially distributed, and for which the service time has an arbitrary (non-negative) distribution with mean ν^{-1}. Then the mean duration of the busy period at the output of the queue is

$$\bar{\tau} = \frac{1}{\nu-\lambda} \qquad , \qquad 0 \leq \lambda < \nu$$

where λ is the expected number of customers per unit time. Not surprisingly, similar arguments [22] can be used to show that the mean time to empty such a queue, given that there are x customers waiting and the service of a customer has just begun, is

$$\bar{\tau} = \frac{x}{\nu-\lambda}$$

In communication terms, if there are x_i^j bits in a queue, and the mean number of bits leaving and entering the queue per second are ν_i^j and λ_i^j respectively, then the expected time before the queue first becomes empty is

$$\bar{\tau}_i^j = \frac{x_i^j}{\nu_i^j-\lambda_i^j}$$

Of central importance here is the parallel to the constant flow plus backlog case: if r_i^j is a constant flow requirement from node i to node j, and $\nu_i^j > r_i^j$ is the capacity allocated to this commodity, then clearly the time required to empty a backlog x_i^j is given by

$$\tau_i^j = \frac{x_i^j}{\nu_i^j - r_i^j}$$

Thus the mathematical problems of finding feasible capacity allocations to minimize

$$\max \quad \tau_i^j \qquad i,j \in \mathcal{N}$$

in the deterministic case, and to minimize

$$\max \quad \overline{\tau_i^j} \qquad i,j \in \mathcal{N}$$

in the random case, are identical. Similarly, there is equivalence also between the stochastic and deterministic problem formulations leading to iterative minimization of the parameter sequence $\alpha_2, \alpha_3, \ldots, \alpha_M$.

The nature of the dynamic flow solution produced by the minimax routing procedure in the stochastic case can be illuminated by recognizing that the solution of the dominant (i.e., first) problem, embodied in the triple $(\tau_1 \ S_1 \ R_1)$, is unchanged even if all requirements for commodities <u>not</u> in R_1 are zero. If we assume that this is so, then all links not in S_1 serve only to convey commodities in R_1 from their nodes of origin to S_1, and thence on to their destinations. The links in S_1, on the other hand, are saturated, so that the aggregate flow of all commodities together thru S_1 is

$$|S_1| \triangleq \sum_{(\ell,k)\in S_1} c_{\ell k} = \sum_{i,j \in R_1} \nu_i^j$$

Moreover, this aggregate flow is apportioned among the commodities in R_1 in such a way that

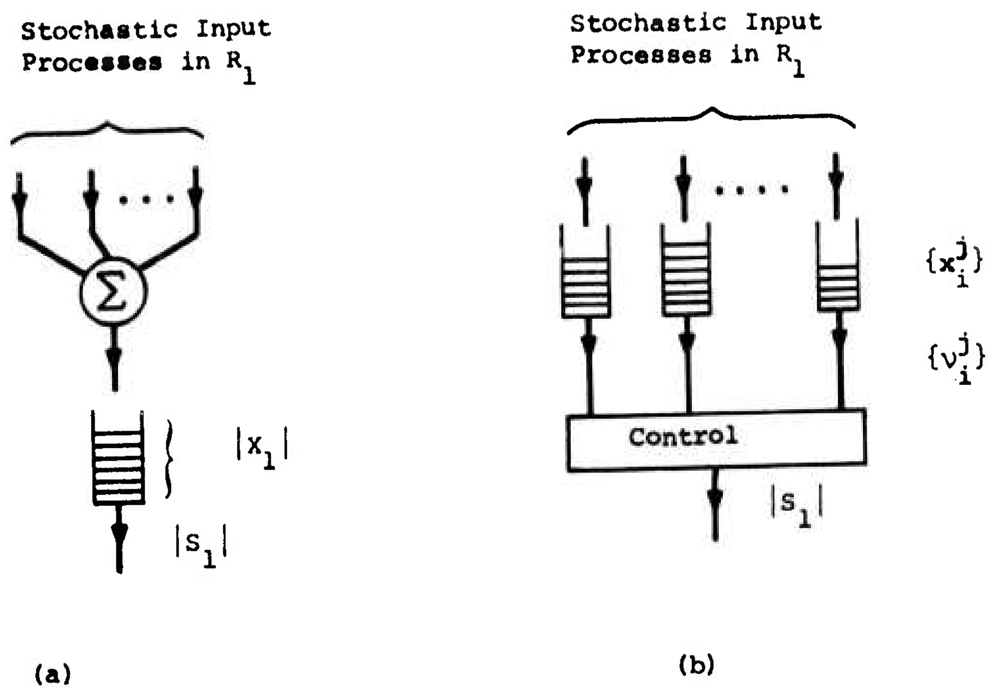
$$\overline{\tau_i^j} = \frac{x_i^j}{\nu_i^j - \lambda_i^j} \triangleq \overline{\tau_1} \quad \text{(a constant) for all } x_i^j \in R_1$$

Summing over all $i,j \in R_1$ yields

$$|x_1| = \overline{\tau}_1(|S_1| - \Lambda_1)$$

where $|x_1| \triangleq \sum x_i^j$ and $\Lambda_1 \triangleq \sum \lambda_i^j$. It follows that under the minimax routing policy the aggregate of the queues in R_1 is depleted exactly as fast as if all the inputs were flowing directly into a single queue with output capacity $|S_1|$, as shown in Figure 6a. Moreover, $|S_1|$ is the maximum capacity that the network $[c_{\ell k}]$ could possible provide for commodities in R_1: no other commodity flows thru S_1, and S_1 forms a cutset disconnecting all source nodes in R_1 from destination nodes in R_1. The power of the minimax policy lies in the fact that, given the network state, it identifies the "worst" congestion problem and alleviates it with maximum resources.

Insofar as assuring equivalence to the aggregate queue of Figure 6a is concerned, however, minimax allocation of $|S_1|$ among the different commodities in R_1 suffices, but is not necessary: it is easy to see in Figure 6b that any control policy $\{\nu_i^j\}$ which keeps S_1 saturated whenever $|x_1| \neq 0$ is equally efficient insofar as minimizing the time by which all x_i^j can be simultaneously emptied is concerned. It can also be shown [22] that all such policies also yield the same mean delay per bit. The main advantage of minimax routing appears to lie in the parallelism between the deterministic and stochastic cases, in the identification of S_1 and R_1, and in the determination of commodity routes that are feasible as well as efficient. From a practical point of view, the equivalence of all flows that keep S_1 saturated means that we can deviate from routes that are continuously readjusted to maintain the minimax property at each time instant, without necessarily incurring a penalty in mean performance.



<u>Figure 6</u>

The consolidated queue in (a), and each of the separate queues in (b), will all become empty simultaneously for any control policy that keeps $|s_1|$ saturated whenever possible.

3.3.2.5 Future Work

Although most of our prospective work in routing will shift in emphasis next year, further exploration seems desirable along two lines of current investigation. Specifically, we are still uncertain as to (i) how the complexity of a minimax routing implementation grows as a function of network size, and (ii) how the solution of the optimum dynamic routing problem is affected when the commodity requirements have multiple destinations.

Our principal new objectives for next year include incorporating network stability and adaptability, in addition to delay, as explicit measures of routing policy desirability. Several new models are now under initial development [19, Sec. IV], their major goal being to accommodate analysis of these properties in the face of dynamic input changes and delays in node response.

Models that come naturally to mind at this juncture involve a two time-constant approach to routing. A system optimizing, quasistatic routing policy can be calculated by a global distributed algorithm based on relatively long-term running estimates of network topology and congestion parameters. We may think of the resulting routing structure as defining a slowly varying network "operating point", around which rapidly fluctuating localized perturbations could be implemented via the exchange of dynamic state information between neighbors. As a first step towards this end, two distributed algorithms have been programmed (but not yet evaluated) which converge for constant input flows onto routes that respectively minimize (i) the average network delay, and (ii) the maximum network link saturation level. Several approaches to control of the fast time constant perturbations bear investigation, one being via a linear-quadratic optimal control theory formulation, and another (for sparse networks) being via an approximate dynamic programming formulation.

An interesting conceptual basis for tying these ideas together grows out of looking at a somewhat stronger type of loop-freedom than is required just for convergence of the distributed routing algorithm itself. Specifically, one now asks not only for freedom from loops in traffic going to a given destination, but also freedom from loops which involve traffic to several destinations, as shown in Figure 7. Mathematically a feasible vector of link flows $\vec{f} = (f_1, \ldots, f_L)$ satisfying a given set of input requirements is said to be loop free if there is no other flow $\vec{f}'$ satisfying the same requirements such that $f_i' \leq f_i$ for $1 \leq i \leq L$, with strict inequality for at least one i. The reason for this definition is that if such an $\vec{f}'$ existed, then $\vec{f} - \vec{f}'$ would be non-zero for no inputs, i.e., a purely looping set of flows.

One way to characterize a loop free flow is as a solution to some flow optimization problem. Another is to recognize that for any loop free $\vec{f}$ there is a vector $\vec{d}$ (whose components are positive link distances) such that $\vec{f}$ minimizes $\vec{f} \cdot \vec{d}$ over the set of feasible flows. This means that all traffic corresponding to $\vec{f}$ takes minimal distance routes according to $\vec{d}$. Typically there will be several different minimal distance routes between two nodes and thus multiple choices of loop free flows $\vec{f}$ corresponding to the same distance vector $\vec{d}$. In other words $\vec{d}$ determines a set of allowable routes, consistent in the sense of not allowing loops, and a given $\vec{f}$ determines the allocation of flow among these allowable routes. It is intriguing, then, to look for routing algorithms which change $\vec{d}$ slowly using more or less global information and which change $\vec{f}$, for fixed $\vec{d}$, more rapidly using local information.

Evaluation of the performance of adaptive routing algorithms, in the face of dynamic changes in traffic requirements and network topology, is one of our major goals for next year. A central obstacle to achieving this goal will be the difficulty of coping with statistical dependencies among alternate routes.

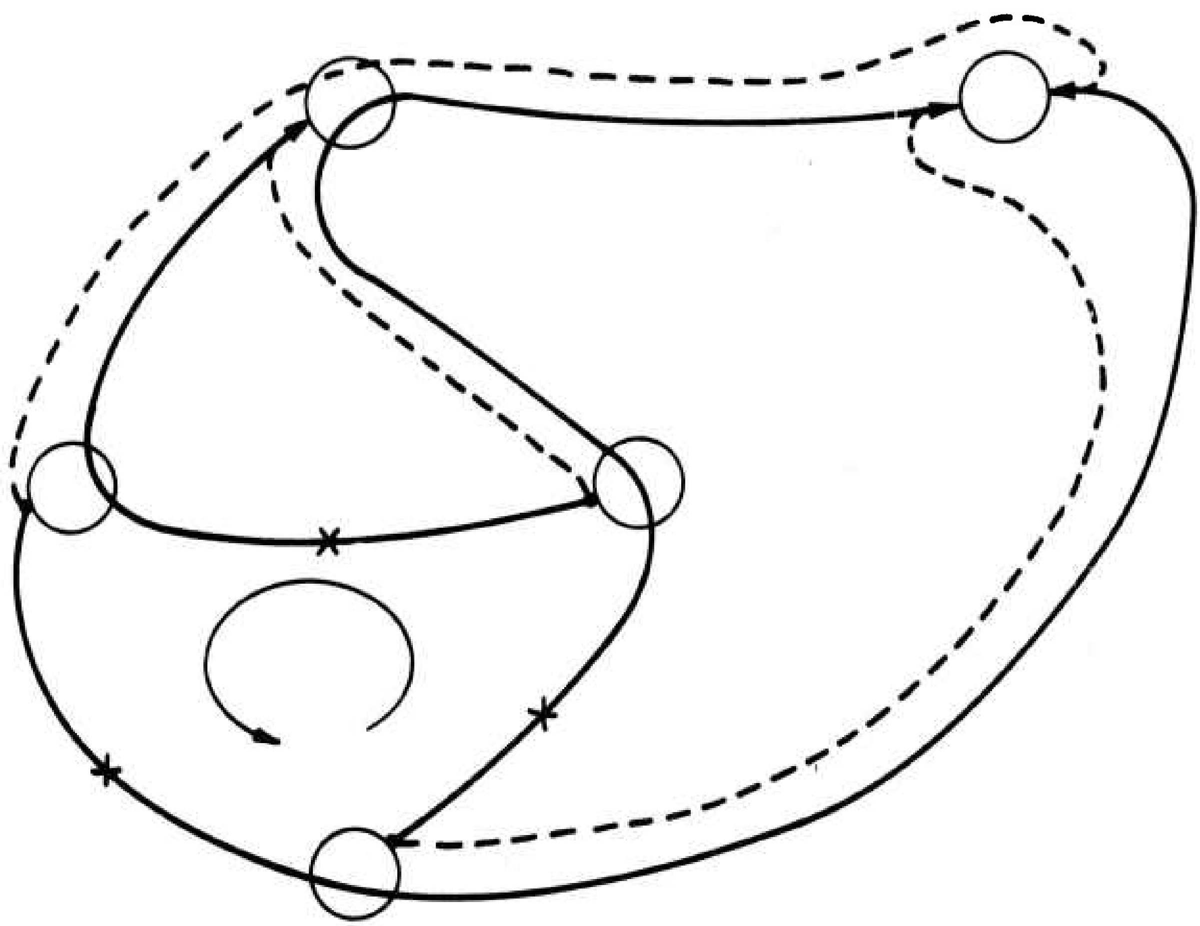


Figure 7 Multicommodity Loop.

If all commodity flow requirements are the same (say 10 bits/sec), then the dashed routing meets the same requirements as the solid routing, but eliminates all flows in the links marked by X's.

3.3.3 Network Topology

Our attempts to come to grips with the problem of network topology thus far have centered on looking for appropriate engineering criteria for prefering one topology to another. By topology we mean those structural aspects that are invariant to geographical parameters such as length. Thus two networks are the same topologically if their node-arc incidence matrices and arc capacities are isomorphic, even though one network might fit into a room and the other span a continent. For simplicity we also make the usual assumption that the arcs (e.g., links) are unidirectional, and that only one link (of arbitrary but finite capacity) joins any two nodes.

One state-of-the-art [23] approach to topological optimization can be described roughly as follows. Assume a fixed total supply of link capacity, and a fixed N-node multicommodity steady-flow requirements matrix. Divide the available capacity among all N(N-1) possible arcs in such a way that the value of an appropriate multicommodity flow objective function is minimized over all feasible flows and all capacity assignments, and call the resulting network "best" for that set of requirements.

Aside from computational issues, the main defect of the foregoing approach lies in the fact that the constraint on aggregate link capacity produces solutions that may be undesirable from a network reliability point of view. A simple example of this is shown in Figure 8, in which it is clear that counting capacity that is bridged around a node differently than we count the same amount of capacity when it is patched thru a node favors topologies with more links of smaller capacity over fewer links having larger capacity.

Exactly the opposite is, of course, preferable in terms of queueing theory. More cogently, the adaptability of topologies with fewer links of higher capacity is superior, in the sense that Figure 8 (i) can be obtained from Figure 8 (iii) by switching at node b, whereas (iii) cannot be obtained from (i). Thus there will be requirements matrices that can be satisfied by (iii) but not by (i). We conclude that incorporating

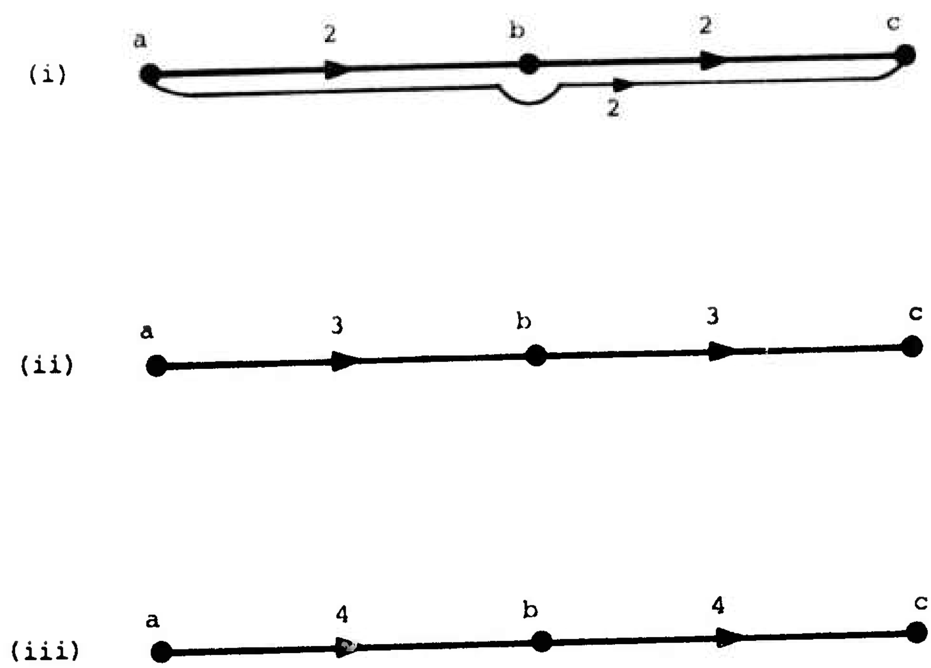


Figure 8 Topological Games

Assume that $r_{ab} = r_{ac} = r_{bc} = 1$ and that our objective is to route traffic so as to minimize

$$\Phi \triangleq \sum_{\ell} \frac{f_\ell}{c_\ell - f_\ell}$$

We then have

$$\Phi = \begin{cases} 3 & \text{, for case (i)} \\ 4 & \text{, for case (ii)} \\ 2 & \text{, for case (iii)} \end{cases}$$

Thus (i) is optimum if (i) and (ii) are considered equivalent in their use of link capacity, whereas (iii) is optimum if (i) and (iii) are considered equivalent.

adaptive routing into our network operating doctrine implies that we should address questions of topology from the viewpoint of the class of all requirements that can be accommodated, rather than for the viewpoint of only a single member of the class.†

3.3.3.1 Future Work

One approach to looking for engineering criteria in the context of adaptive networks lies in considering costs to be associated with exploiting whatever adaptive capabilities a particular capacitated topology affords, rather than associating cost with the capacities themselves. An example of this approach in the case of a trunking network connecting telephone central offices would be to minimize the total cost of the switch plant, rather than the total number of trunks. Here "costs" are of two kinds; first, the complexity of calculating routes to satisfy different requirements; and second, the complexity of implementing those routes within the network.

We anticipate that two investigations now in progress will provide insight along these lines. The first [24] is considering multiaccess broadcast channels of the slotted Aloha type. The problem here is to find algorithms to resolve conflicts caused when several sources transmit simultaneously. Although a number of previous algorithms have been unstable, a new class of algorithms has been devised, and proved stable. Additional properties of these algorithms are being investigated, and an attempt is also being made to obtain (for purposes of comparison) an information theoretic upper bound on the performance of all strategies to resolve multiaccess conflict. Aloha channels are of interest in their own right, but they may also be viewed as an extremal topology (with no structure at all) in which the cost of "implementing routes" is negligible relative to the cost of "calculating routes".

† The foregoing discussion tacitly assumes that network connectivity is great enough to cope with link or nodal failures.

The second investigation will seek to understand routing behavior for a "network of networks". The complexity of optimum routing calculations clearly grows out of hand as networks become arbitrarily large, unless some structure is imposed. The interesting topological question is the tradeoff obtainable between routing costs and routing performance, as a function of overall size and the choice of network and subnetwork structure. Both distributed and centralized algorithms deserve study.

An entirely different aspect of topology concerns network sensitivity to parameter changes, such as degradations in link capacity. Related work [25] has been done on transportation networks; specifically, for freeway corridor nets the change in steady-state optimal routing strategy due to an imposed change in arriving flow was calculated. The change in arriving flow could be due to a traffic accident somewhere in or just outside of the network, or due to some other cause. For special classes of networks, conditions were found under which perturbations caused by accidents have a significant effect on only a limited region of the network.

This work, if it can be suitably extended, would be of value in communications networks. The region significantly affected by a given incident can be isolated by means of the analysis, which involves only a small number of multiplications of relatively small matrices. We expect to investigate the usefulness of the technique both for analyzing the reliability of proposed networks, and for discovering the weak points of existing networks. An additional side benefit would be knowing the change in static optimal routing strategy due to imposed parameter shifts, which should be helpful in dynamic message routing.

3.4 Communication Links

3.4.1 Protocol Data

A number of results have been achieved in the past year concerning protocols for representing message lengths and addresses on communication links. This work stems from earlier results [26] which provided tight upper and lower bounds on the number of bits required by any such protocol in the limit of a very large number of virtual paths in the network.

One investigation [27] considered various strategies for specifying the length of messages in a network. One strategy is to break messages into equal length packets, with a variable length final packet. All but the last packet of a message are prefixed with a zero, and the final packet is prefixed with a one followed by an encoding of the final packet length. If the message lengths are geometrically distributed and the fixed packet length is properly chosen (roughly $(\ln 2)^{-1}$ times the average message length), then this technique turns out to be equivalent to generating the (optimal) Huffman source code for the message length.

A second strategy is to use a flag consisting of a zero followed by some number k of ones to delimit the end of a message. To avoid the spurious appearance of this flag within the message, each zero followed by k-1 ones in the message has a zero inserted after the k-1 ones by the transmitter and deleted by the receiver. This strategy is similar to a standard strategy used in the IBM's SDLC and a number of other systems which use the same flag followed by an unneeded zero. This strategy, rather surprisingly, requires only about 1/2 bit per message, on the average, more protocol bits than the optimal packet strategy described above.

A more important problem concerns the interaction between queueing delays and the protocols used to represent message addresses and lengths. It turns out to be possible to represent the addressing function with fewer protocol bits when queues are long than when they are short, and this can be used to help counteract the instability in queue lengths caused by congestion. One class of strategies with the above behaviour has been analyzed [28, 29] and can be roughly characterized as statistical multiplexing. Mathematically the problem is equivalent to that of round robin queues with changeover times. The changeover times correspond to the protocol necessary to encode the number of messages in each queue when the queue is served.

One of the basic stumbling blocks in analyzing queueing delays on the links of a network is the lack of appropriate bounds and approximations on queue length behaviour. One partial result in this direction is a new lower bound, extending Kingman's results, on the expected waiting time in a G/G/1 queue [30].

3.4.1.1 Future Work

One of the basic goals in our work on protocols and queueing delays is to develop lower bounds on protocol and queue lengths which are independent of the particular strategy for representing the protocols. Since information (in the mathematical sense) is invariant to representation, an obvious approach is to study the queueing of information and the flow of information in a network rather than the queueing and flow of the binary digits used to represent the information. There are fundamental difficulties in doing this, partly due to the multiple source nature of the information involved and partly due to the need to use average measures of information in a dynamic way. One of our goals for next year is to fully understand and hopefully to overcome these obstacles.

3.4.2 Error Control

Error detection and retransmission protocols have been the subject of another investigation [31]. One important result of this work is a technique for proving the correctness of such protocols. Error detection protocols, involving a two way interaction between two nodes, form the simplest non-trivial case of network situations where deadlocks are a major danger. Another result is the development of a fixed block length error detection strategy which uses no bits for message acknowledgement other than the check bits of the block code used for error detection. The elimination of one or two bits of acknowledgement overhead is not of great practical significance, but it does have theoretical importance. A final result is the development of a strategy for variable block length messages which requires one bit of acknowledgement overhead per message plus extra bits when errors occur.

3.4.3 Estimation

In adaptive network control one is often interested in parameters of the network behavior which are not directly observable. For example, one such parameter is the derivative of the delay on a link with respect to the flow over that link. To see the importance of this derivative, let us denote by $D_{ik}(f_{ik})$ the total delay faced by all messages passing through link (i,k) per unit time, where f_{ik} is the flow (in messages/sec.) through this link. Then the total delay over the network per unit time is

$$D_t = \sum D_{ik}(f_{ik})$$

Suppose now that some nominal flows $\tilde{f}_{ik}$ exist in the network satisfying some nominal flow requirements $\{\tilde{r}_i^j,\ i,\ j \in \mathcal{N},\ i \neq j\}$ and that the flow requirement from some node i to some destination j increases by an incremental amount δr_i^j. The question is what path should be chosen for this extra traffic. The "user-optimized" answer to this question (which is the one implemented in the ARPANET) is to choose the path over which the total delay is minimal. Clearly such a path will be the best for the extra traffic δr_i^j itself, but it disregards the fact that this choice may hurt everybody else, i.e., the existing traffic. If the quantity to be optimized is indeed the average delay (which is proportional to D_t) both effects must be taken into consideration: the delay incurred by the extra traffic itself, as well as the extra-delay suffered by the existing traffic. This is called "system optimization" and can be done by observing that if one chooses a path P from i to j for the extra traffic, then the extra total delay δD_t will be (up to first order)

$$\delta D_t = \sum_{(\ell,m)\in P} \frac{\partial D_t}{\partial f_{\ell m}} \cdot \delta r_i^j = \sum_{(\ell,m)\in P} \frac{dD_{\ell m}}{df_{\ell m}} \cdot \delta r_i^j$$

which essentially says that the flow in each of the links of the chosen path will increase by δr_i^j.

The following decentralized routing algorithm is suggested by this expression:

a) estimate the incremental de. y over each link in the network (the estimation c done locally and the procedure will be described presently) and

b) use these quantities to update the routing tables essentially in the same way the estimated delay is used in ARPANET.

Other strategies can be designed as well, depending on the particular network under design. In [18] a recursive algorithm has been proposed to divide the traffic in an optimal way over each of the outgoing links, so that the total delay will be minimized. Also, in a network controlled from a central site, the router can periodically collect the estimated incremental delays and use them to find the gradient of the delay, the projection of which on the flow requirement subspace will provide the steepest descent direction for change of the flows. Such a strategy has been proposed in [32].

No matter which of the strategies indicated above is used, the point is that all methods need as a basic quantity the incremental delay $dD_{\ell m}/df_{\ell m}$. One way to find it is using some queueing formula, but such an approach will necessarily involve a certain number of assumptions, which one would like to avoid if possible. It is therefore of importance to estimate the incremental delay directly, thus reducing the dependence of the algorithms on various assumptions. In fact it will be seen presently that the only necessary assumption for the estimation algorithms to make sense is stationarity over the invervals between routing changes.

The procedure we propose for estimating the derivative of the delay over a given link is based on "imagining" that the flow over the link is changed by an incremental amount and evaluating what effect this hypothesized change would have on the total delay of the messages [33]. For example, assume (hypothetically) that each packet arriving at the link will be transmitted with probability $(1-\varepsilon)$ and eradicated with probability ε, independently from packet to packet. The effective rate will then be reduced in the average by

$$\delta f = \frac{M\varepsilon}{T}$$

where

T = period of interest over which the estimate is calculated

M = number of arrivals during T.

Also, the probability of removing two or more packets at a time is of order ε^2 and therefore of second order in δf, so that one needs consider only the effect of removing one packet at a time. It is also easy to see that removing a packet from one given busy period has no effect on packets served in other busy periods, so that one only needs to consider the effect of the removal on packets from the same busy period.

For a given busy period, let c_m^n be the amount of system time that the m-th packet would save if the n-th packet were to be removed. If we denote by d_n and a_n respectively the departure and arrival time of the n-th packet relative to the beginning of the busy period, then one can easily obtain the following recursive formulas:

$$c_m^n = 0 \qquad \text{for } m < n$$

$$c_m^n = d_n - a_n \qquad \text{for } m = n$$

$$c_{n+1}^n = d_n - \max\,(a_{n+1}, d_{n-1}) \text{ where we take } d_0 = 0$$

and for $m > n+1$

$$c_m^n = \min\,(c_{m-1}^n, d_{m-1} - a_m)$$

We assume here a first-come first-serve discipline.

The foregoing equations each hold, of course, for all packets from the same busy period. Over B busy periods, each containing respectively $\{N_i, i = 1,\ldots, B\}$ packets, the total effect will therefore be

$$\delta D = \frac{1}{T} \sum_{i=1}^{B} \sum_{n=1}^{N_i} \sum_{m=n}^{N_i} \varepsilon \, c_m^n$$

and the desired estimate $\delta D/\delta f$ of the derivative will be

$$\frac{\delta D}{\delta f} = \frac{\sum_{i=1}^{B} \sum_{n=1}^{N_i} \sum_{m=n}^{N_i} c_m^n}{\sum_{i=1}^{B} N_i}$$

A slightly different algorithm is to add a new hypothetical packet and calculate its influence on the total delay.

Several analytical and numerical investigations have been done [33] regarding the performance of the proposed algorithms. Unbiasedness of one of the algorithms for M/D/1 queues has been proven analytically, and there is strong analytical evidence that the second algorithm is also unbiased, although no full proof is available yet. Monte Carlo simulations for M/D/1, M/M/1, and D/M/1 queues have shown good performance of the algorithms in terms of their bias and efficiency. In addition, the recursive form of the proposed algorithm indicates that its complexity is relatively small, although no full investigation has been performed yet.

3.4.3.1 Future Work

Since the estimation procedure must be performed at a lower priority level than the actual data transmission, it is very important that its storage and computational requirements not be too high. Further investigation into these requirements, as well as into the performance of the algorithms for various queue statistics and line protocols, is therefore necessary.

The proposed algorithms assume no statistical knowledge of the queueing process. In certain situations, however, some statistical properties are available, which if taken into account can lead to better and more efficient algorithms. It is therefore important to develop alternate algorithms that take these properties into account.

References

[1]* Segall, A., "Dynamic File Assignment in a Computer Network", IEEE Trans. Autom. Control, Vol. AC-21, No. 2, pp. 161-173, April 1976.

[2]* Segall, A. and Sandell, Jr., N.R., "Dynamic File Assignment in a Computer Network, Part II: Decentralized Control", to be submitted to IEEE Trans. Autom. Control.

[3]* Ros-Perán, F.A. and Segall, A., "Dynamic File Assignment in a Computer Network, Part III: Multiple Copies and Failures, to be submitted to IEEE Trans. on Comm.

[4]* Ros-Perán, F.A., "Dynamic File Allocation in a Computer Network", Electronic System Laboratory Report ESL-R-667, M.I.T., June 1976.

[5]* Wunderlich, E.F., "Load Sharing in a Computer-Communication Network" Electronic Systems Laboratory Report ESL-R-678, M.I.T., August 1976.

[6]* Wunderlich, E.F., Defenderfer, J.E., and Wozencraft, J.M., "Statistical Load Sharing in a Computer-Communication Network", to be submitted for publication.

[7] Bowdon, E.K., Sr., "Dispatching in Network Computers", Proc. Symposium on Computer-Communication Networks and Teletraffic, Brooklyn, Polytechnic Press, 1972.

[8] McGregor, P. and Boorstyn, R.R., "Load Sharing in a Computer Network", International Conference on Communications, San Francisco, June 1975.

[9] Cantor, D.G. and Gerla, M., "Optimal Routing in a Packet Switched Computer Network", IEEE Trans. on Computers, Vol. C-23, Oct. 1974.

[10] McQuillan, J.M., et al, "Improvements in the Design and Performance of the Arpa Network", Proc. FJCC, 1972.

[11] Cerf, V. and Kahn, R., "A Protocol for Packet Network Intercommunication", IEEE Trans. on Comm. Vol. COM-22, May 1974.

[12] Davies, D.W., "The Control of Congestion in Packet Switching Networks", IEEE Trans. on Comm. Vol. COM-20, Jan. 1972.

---

*Papers prepared in conjunction with this project.

[13] Chou, W. and Gerla, M., "A Unified Flow Control Strategy for Computer Communications", 4th Data Communications Symposium, Quebec, Canada, Oct. 1975.

[14] Merlin, P.M. and Farber, D.J., "On the Recovery of Communication Protocols", Proceedings of ICC, Phila., June 1976.

[15] Kleinrock, L., Communication Nets; Stochastic Message Flow and Delay, McGraw-Hill 1964.

[16]* Defenderfer, J.E., "Shortest Route Algorithms for Sparsely Connected Networks", to be submitted for publication.

[17]* Defenderfer, J.E., "Comparative Analysis of Routing Algorithms for Computer Networks", ScD. Thesis Proposal, M.I.T., June 1976.

[18]* Gallager, R.G., "An Optimal Routing Algorithm Using Distributed Computation", IEEE Trans. on Comm., special issue on Computer Networks, 1976.

[19]* Segall, A., "The Modeling of Routing in Data-Communication Networks", IEEE Trans. on Comm., special issue on Computer Networks, 1976.

[20]* Moss, F.H., "Optimal Control Methods in the Design of Routing in Data-Communication Networks, Ph.D. Thesis, in preparation, Oct. 1976.

[21]* Wozencraft, J.M. and Yee, J.R., "Minimax Routing", to be submitted for publication.

[22]* Yee, J.R., ScD Thesis Proposal, M.I.T., in preparation.

[23] Kleinrock, L., Queueing Systems, Vol. 2: Computer Applications, J. Wiley, New York 1976.

[24]* Capetanakis, J., "Multiple Access Broadcast Channels", Ph.D. Thesis Proposal, M.I.T., in preparation.

[25] Dersin, P., Gershwin, S.B., and Athans, M., "Sensitivity Analysis of Optimal Static Traffic Assignments in a Large Freeway Corridor, Using Modern Control Theory", Electronic Systems Laboratory Report ESL-R-671, M.I.T., July 1976.

---

*Papers prepared in conjunction with this project.

[26]* Gallager, R.G., "Basic Limits on Protocol Information in Data Communication Networks", IEEE Trans. Infor. Theory, July 1976.

[27]* Camrass, R.J., "Protocol Problems Associated with Simple Communication Networks", Electronic System Laboratory Report ESL-R-673, M.I.T., Jan. 1976.

[28]* Humblet, P., "Multiple Input Single Server Queue with Application to Communication Concentrators", to be submitted for publication.

[29]* Humblet, P., "Optimal Source Coding for a Class of Integer Alphabets", to be submitted for publication.

[30]* Humblet, P., "A Lower Bound for the Average Waiting Time in a G/G/1 Queue", to be submitted for publication.

[31]* Golestaani, S.J., "Design of a Retransmission Strategy for Error Control in Data Communication Networks", Electronic System Laboratory Report ESL-R-674, M.I.T., May 1976.

[32] Schwartz M. and Cheung, C.K., "The Gradient Projection Algorithm for Multiple Routing in Message-Switched Networks", IEEE Trans. on Comm., Vol. COM-24, April 1976.

[33]* Bello, M., "Estimation of Incremental Delay for Packet-Switched Data Networks, S.M. Thesis, in preparation, Sept. 1976.

---

*Papers prepared in conjunction with this project.

Distribution List

Defense Documentation Center Cameron Station Alexandria, Virginia 22314	12 copies
Assistant Chief for Technology Office of Naval Research, Code 200 Arlington, Virginia 22217	1 copy
Office of Naval Research Information Systems Program Code 437 Arlington, Virginia 22217	2 copies
Office of Naval Research Code 1021P Arlington, Virginia 22217	6 copies
Office of Naval Research Branch Office, Boston 495 Summer Street Boston, Massachusetts 02210	1 copy
Office of Naval Research Branch Office, Chicago 536 South Clark Street Chicago, Illinois 60605	1 copy
Office of Naval Research Branch Office, Pasadena 1030 East Green Street Pasadena, California 91106	1 copy
New York Area Office (ONR) 715 Broadway - 5th Floor New York, New York 10003	1 copy
Naval Research Laboratory Technical Information Division, Code 2627 Washington, D.C. 20375	6 copies

Dr. A. L. Slafkosky Scientific Advisor Commandant of the Marine Corps (Code RD-1) Washington, D.C. 20380	1 copy
Office of Naval Research Code 455 Arlington, Virginia 22217	1 copy
Office of Naval Research Code 458 Arlington, Virginia 22217	1 copy
Naval Electronics Laboratory Center Advanced Software Technology Division Code 5200 San Diego, California 92152	1 copy
Mr. E. H. Gleissner Naval Ship Research & Development Center Computation and Mathematics Department Bethesda, Maryland 20084	1 copy
Captain Grace M. Hopper NAICOM/MIS Planning Branch (OP-916D) Office of Chief of Naval Operations Washington, D.C. 20350	1 copy
Mr. Kin B. Thompson Technical Director Information Systems Division (OP-91T) Office of Chief of Naval Operations Washington, D.C. 20350	1 copy
Advanced Research Projects Agency Information Processing Techniques 1400 Wilson Boulevard Arlington, Virginia 22209	1 copy